AMENDMENTS TO THE CLAIMS

1. (Currently amended) In a computer system, a method for retrieving events from an event port, the method comprising:

receiving from a computer software application, a request to retrieve a specified number of events from an event port to which completed events are posted by one or more event sources;

determining whether the specified number of events is available at the event port;

if the specified number of events is available at the event port, retrieving the specified number of events from the event port and returning the retrieved events to the requesting computer software application; [[and]]

if fewer events than the specified number of events are available at the event port, placing the request in a request queue with requests to be processed at a later time and ordering the request queue based on priorities of the requests in the request queue; and

changing the priorities of the requests in the request queue based on the number of events available at the event port, wherein the changing is further based on a specified number of events to be retrieved as part of at least one request received in response to the number of events available at the event port.

2. (Original) The method of claim 1, wherein ordering comprises:

placing requests with a higher priority ahead of requests with lower priority in the request queue.

3. (Original) The method of claim 1, wherein ordering comprises:

placing two or more requests with a same priority in a stack; and

placing the stack in the request queue based on the priority of the requests in the stack.

4. (Original) The method of claim 1, wherein the specified number of events to be retrieved from the event port indicates a priority of the request.

5. (Original) The method of claim 4, wherein a priority of a request is inversely proportional to the specified number of events.

6. (Original) The method of claim 1, wherein the request queue contains requests generated by one or more computer software application threads.

7. (Original) The method of claim 1, wherein the request queue contains requests generated by one or more computer software application processes.

8. (Original) The method of claim 1, wherein the number of events to be retrieved from the event port is specified by the computer software application.

9. (Original) The method of claim 8, wherein the number of events to be retrieved from the event port is specified by the computer software application based on user input.

10. (Original) The method of claim 1, further comprising:

if fewer events than the specified number of events are available at the event port, determining whether there are any requests in the request queue that can be satisfied by the available number of events at the event port; and

if there are requests in the request queue that can be satisfied, retrieving the specified number of events from the event port for one or more such requests and returning the retrieved events to the requesting computer software application.

11. (Original) The method of claim 1, wherein the request has an associated timeout prior to which the request must be satisfied.

12. (Original) The method of claim 11, further comprising:

if a timeout occurs for a request while the request is in the request queue, retrieving all the available events at the event port at the time of timeout; and

returning the request to the computer software application with the retrieved events.

13. (Original) The method of claim 1, further comprising:

returning an empty request to the requesting software application if the request cannot be satisfied.

14. (Original) The method of claim 13, wherein returning comprises:

returning the empty request together with an error code indicating the cause for why the request cannot be satisfied.

15. (Original) The method of claim 1, further comprising:

returning an empty request to the requesting software application if one or more of the following error conditions occur:

the request contains an invalid event port identifier, an event or a list of events list cannot be delivered, a timeout argument is out of range, and a timeout interval expires before an expected number of events has been posted to the event port.

16. (Original) The method of claim 1, wherein returning the retrieved events to the requesting computer software application comprises returning one or more of:

one or more detected events, one or more event source identifiers where the detected events were generated, one or more objects specific to an event source, and one or more user defined values.

17. (Original) The method of claim 1, further comprising:

if the specified number of events is zero, identifying the number of available events at the event port; and

informing the requesting computer software application of how many events are available at the event port.

18. (Original) The method of claim 1, wherein the events are asynchronous events.

19. (Original) The method of claim 1, wherein the events are transaction events.

20. (Original) The method of claim 1, wherein the event sources include one or more of: input devices, output devices, timers, signals, file updates, applications, system libraries, and drivers.

21. (Currently amended) A computer program product, stored on a machine-readable medium, comprising instructions operable to cause a computer to:

receive from a computer software application, a request to retrieve a specified number of events from an event port to which completed events are posted by one or more event sources;

determine whether the specified number of events is available at the event port;

if the specified number of events is available at the event port, retrieve the specified number of events from the event port and returning the retrieved events to the requesting computer software application; [[and]]

if fewer events than the specified number of events are available at the event port, place the request in a request queue with requests to be processed at a later time and order the request queue based on priorities of the requests in the request queue; and

changing the priorities of the requests in the request queue based on the number of events available at the event port, wherein the changing is further based on a specified number of events to be retrieved as part of at least one request received in response to the number of events available at the event port.

22. (Original) The computer program product of claim 21, wherein the instructions to order comprise instructions to:

place requests with a higher priority ahead of requests with lower priority in the request queue.

23. (Original) The computer program product of claim 21, wherein the instructions to order comprise instructions to:

place two or more requests with a same priority in a stack; and

place the stack in the request queue based on the priority of the requests in the stack.

24. (Original) The computer program product of claim 21, wherein the specified number of events to be retrieved from the event port indicates a priority of the request.

25. (Original) The computer program product of claim 24, wherein a priority of a request is inversely proportional to the specified number of events.

26. (Original) The computer program product of claim 21, wherein the request queue contains requests generated by one or more computer software application threads.

27. (Original) The computer program product of claim 21, wherein the request queue contains requests generated by one or more computer software application processes.

28. (Original) The computer program product of claim 21, wherein the number of events to be retrieved from the event port is specified by the computer software application.

29. (Currently amended) The computer program product of claim [[29]]28, wherein the number of events to be retrieved from the event port is specified by the computer software application based on user input.

30. (Original) The computer program product of claim 21, further comprising instructions to:

determine whether there are any requests in the request queue that can be satisfied by the available number of events at the event port, if fewer events than the specified number of events are available at the event port; and

retrieve the specified number of events from the event port for one or more such requests and returning the retrieved events to the requesting computer software application, if there are requests in the request queue that can be satisfied.

31. (Original) The computer program product of claim 21, wherein the request has an associated timeout prior to which the request must be satisfied.

32. (Original) The computer program product of claim 31, further comprising instructions to:

retrieve all the available events at the event port at the time of timeout if a timeout occurs for a request while the request is in the request queue; and

return the request to the computer software application with the retrieved events.

33. (Original) The computer program product of claim 21, further comprising instructions to:

return an empty request to the requesting software application if the request cannot be satisfied.

34. (Original) The computer program product of claim 33, wherein the instructions to return comprise instructions to:

return the empty request together with an error code indicating the cause for why the request cannot be satisfied.

35. (Original) The computer program product of claim 21, further comprising instructions to:

return an empty request to the requesting software application if one or more of the following error conditions occur:

the request contains an invalid event port identifier, an event or a list of events list cannot be delivered, a timeout argument is out of range, and a timeout interval expires before an expected number of events has been posted to the event port.

36. (Original) The computer program product of claim 21, wherein the instructions to return the retrieved events to the requesting computer software application comprise instructions to return one or more of:

one or more detected events, one or more event source identifiers where the detected events were generated, one or more objects specific to an event source, and one or more user defined values.

37. (Original) The computer program product of claim 21, further comprising instructions to:

identify the number of available events at the event port if the specified number of events is zero; and

inform the requesting computer software application of how many events are available at the event port.

38. (Original) The computer program product of claim 21, wherein the events are asynchronous events.

39. (Original) The computer program product of claim 21, wherein the events are transaction events.

40. (Original) The computer program product of claim 21, wherein the event sources include one or more of: input devices, output devices, timers, signals, file updates, applications, system libraries, and drivers.

41. (Original) A queue management apparatus for retrieving transaction events generated by one or more event sources in a computer system, comprising:

an event queue for receiving transaction events generated by one or more event sources, the event queue being accessible through an event port;

a request queue for holding requests to retrieve transaction events from the event queue, each request having an associated priority determining a place of the request in the request queue; and

a queue manager operable to:

receive requests from a computer software application;

organize the received requests in the request queue;

determine whether a sufficient number of events are available in the event queue to fill a request with a highest priority in the request queue; [[and]]

if a sufficient number of events is available in the event queue, retrieve the events needed to fill the request with the highest priority in the request queue through the event port and return the filled request to the requesting computer software application; and

change the associated priorities of the requests in the request queue based on the number of events available in the event queue, wherein the changing is further based on a specified number of events to be retrieved as part of at least one request received in response to the number of events available in the event queue.

42. (Original) The queue management apparatus of claim 41, wherein the request queue includes one or more stacks, each stack containing two or more requests with a same priority, the stacks being ordered in the request queue based on the priority of the requests in the respective stacks.

43. (Original) The queue management apparatus of claim 41, wherein the priority of a request is indicated by a number of events to be retrieved by the request.

44. (Original) The method of claim 41, wherein a request in the request queue has an associated timeout prior to which the request must be satisfied.